Finding, Hitting and Packing Cycles in Subexponential Time on Unit Disk Graphs*†

Fedor V. Fomin[1], Daniel Lokshtanov[2], Fahad Panolan[3], Saket Saurabh[4], and Meirav Zehavi[5]

1 **Department of Informatics, University of Bergen, Bergen, Norway**
fomin@ii.uib.no
2 **Department of Informatics, University of Bergen, Bergen, Norway**
daniello@ii.uib.no
3 **Department of Informatics, University of Bergen, Bergen, Norway**
fahad.panolan@ii.uib.no
4 **Department of Informatics, University of Bergen, Bergen, Norway; and The Institute of Mathematical Sciences, HBNI, Chennai, India**
saket@imsc.res.in
5 **Department of Informatics, University of Bergen, Bergen, Norway**
meirav.zehavi@ii.uib.no

Abstract

We give algorithms with running time $2^{\mathcal{O}(\sqrt{k}\log k)} \cdot n^{\mathcal{O}(1)}$ for the following problems. Given an n-vertex unit disk graph G and an integer k, decide whether G contains

- a path on exactly/at least k vertices,
- a cycle on exactly k vertices,
- a cycle on at least k vertices,
- a feedback vertex set of size at most k, and
- a set of k pairwise vertex-disjoint cycles.

For the first three problems, no subexponential time parameterized algorithms were previously known. For the remaining two problems, our algorithms significantly outperform the previously best known parameterized algorithms that run in time $2^{\mathcal{O}(k^{0.75}\log k)} \cdot n^{\mathcal{O}(1)}$. Our algorithms are based on a new kind of tree decompositions of unit disk graphs where the separators can have size up to $k^{\mathcal{O}(1)}$ and there exists a solution that crosses every separator at most $\mathcal{O}(\sqrt{k})$ times. The running times of our algorithms are optimal up to the $\log k$ factor in the exponent, assuming the Exponential Time Hypothesis.

1998 ACM Subject Classification F.2.2 [Nonnumerical Algorithms and Problems] Geometrical Problems and Computations, G.2.2 Graph Algorithms

Keywords and phrases Longest Cycle, Cycle Packing, Feedback Vertex Set, Unit Disk Graph, Parameterized Complexity

Digital Object Identifier 10.4230/LIPIcs.ICALP.2017.65

* The full version of the paper can be found at arXiv [21], https://arxiv.org/abs/1704.07279.

† Supported by Pareto-Optimal Parameterized Algorithms, ERC Starting Grant 715744, Parameterized Approximation, ERC Starting Grant 306992, and Rigorous Theory of Preprocessing, ERC Advanced Investigator Grant 267959.

44th International Colloquium on Automata, Languages, and Programming (ICALP 2017).
Editors: Ioannis Chatzigiannakis, Piotr Indyk, Fabian Kuhn, and Anca Muscholl;
Article No. 65; pp. 65:1–65:15

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

Unit disk graphs are the intersection graphs of unit circles in the plane. That is, given n-unit circles in the plane, we have a graph G where each vertex corresponds to a circle such that there is an edge between two vertices when the corresponding circles intersect. Unit disk graphs form one of the most well studied graph classes in computational geometry because of their use in modelling optimal facility location [40] and broadcast networks such as wireless, ad-hoc and sensor networks [25, 33, 42]. These applications have led to an extensive study of NP-complete problems on unit disk graphs in the realms of computational complexity and approximation algorithms. We refer the reader to [10, 18, 29] and the citations therein for these studies. However, these problems remain hitherto unexplored in the light of parameterized complexity with exceptions that are few and far between [1, 8, 23, 32, 38].

In this paper we consider the following basic problems about finding, hitting and packing cycles on unit disk graphs from the viewpoint of parameterized algorithms. For a given graph G and integer k,

- Exact k-Cycle asks whether G contains a cycle on exactly k vertices,
- Longest Cycle asks whether G contains a cycle on at least k vertices,
- Feedback Vertex Set asks whether G contains a vertex set S of size k such that the graph $G \setminus S$ is acyclic, and
- Cycle Packing asks whether G contains a set of k pairwise vertex-disjoint cycles.

Along the way, we also study Longest Path (decide whether G contains a path on exactly/at least k vertices) and Subgraph Isomorphism (SI). In SI, given *connected* graphs G and H on n and k vertices, respectively, the goal is to decide whether there exists a subgraph in G that is isomorphic to H. We also assume that a unit disk graph is given by a set of n points in $\mathbb{R}^2$ and there is a graph where vertices correspond to the points and there is an edge between two vertices if and only if the distance between the two points is at most 2.

In parameterized complexity each of these problems serves as a testbed for development of fundamental algorithmic techniques such as color-coding [2], the polynomial method [34, 35, 41, 4], matroid based techniques [20] for Longest Path and Longest Cycle, and kernelization techniques for Feedback Vertex Set [39]. We refer to [12] for an extensive overview of the literature on parameterized algorithms for these problems. For example, the fastest known algorithms solving Longest Path are the $1.66^k \cdot n^{\mathcal{O}(1)}$ time randomized algorithm of Björklund et al. [4], and the $2.597^k \cdot n^{\mathcal{O}(1)}$ time deterministic algorithm of Zehavi [43]. Moreover, unless the Exponential Time Hypothesis (ETH) of Impagliazzo, Paturi and Zane [30] fails, none of the problems above can be solved in time $2^{o(k)} \cdot n^{\mathcal{O}(1)}$ [30].

While all these problems remain NP-complete on planar graphs, substantially faster – *subexponential* – parameterized algorithms are known on planar graphs. In particular, by combining the bidimensionality theory of Demaine et al. [13] with efficient algorithms on graphs of bounded treewidth [17], Longest Path, Longest Cycle, Feedback Vertex Set and Cycle Packing are solvable in time $2^{\mathcal{O}(\sqrt{k})}n^{\mathcal{O}(1)}$ on planar graphs. The parameterized subexponential "tractability" of such problems can be extended to graphs excluding some fixed graph as a minor [15]. The bidimensionality arguments cannot be applied to Exact k-Cycle and this was one of the motivations for developing the new pattern-covering technique, which is used to give a randomized algorithm for Exact k-Cycle running in time $2^{\mathcal{O}(\sqrt{k}\log^2 k)}n^{\mathcal{O}(1)}$ on planar and apex-minor-free graphs [19]. The bidimensionality theory was also used to design (efficient) polynomial time approximation scheme ((E)PTAS) [14, 22] and polynomial kernelization [24] on planar graphs.

It would be interesting to find generic properties of problems, similar to the theory of bidimensionality for planar-graph problems, that could guarantee the existence of subexponential

parameterized algorithms or (E)PTAS on geometric classes of graphs, such as unit disk graphs. The theory of (E)PTAS on geometric classes of graphs is extremely well developed and several methods have been devised for this purpose. This includes methods such as shifting techniques, geometric sampling and bidimensionality theory [29, 27, 26, 28, 11, 37, 23]. However, we are still very far from a satisfactory understanding of the "subexponential" phenomena for problems on geometric graphs. We know that some problems such as Independent Set and Dominating Set, which are solvable in time $2^{\mathcal{O}(\sqrt{k})}n^{\mathcal{O}(1)}$ on planar graphs, are W[1]-hard on unit disk graphs and thus the existence of an algorithm of running time $f(k) \cdot n^{\mathcal{O}(1)}$ is highly unlikely for any function f [36]. The existence of a vertex-linear kernel for some problems on unit disk graphs such as Vertex Cover [9] or Connected Vertex Cover [32] combined with an appropriate separation theorem [1, 8, 38] yields a parameterized subexponential algorithm. A subset of the authors of this paper used a different approach based on bidimensionality theory to obtain subexponential algorithms of running time $2^{\mathcal{O}(k^{0.75}\log k)} \cdot n^{\mathcal{O}(1)}$ on unit disk graphs for Feedback Vertex Set and Cycle Packing in [23]. No parameterized subexponential algorithms on unit disk graphs for Longest Path, Longest Cycle, and Exact k-Cycle were known prior to our work.

Our Results. We design subexponential parameterized algorithms, with running time $2^{\mathcal{O}(\sqrt{k}\log k)} \cdot n^{\mathcal{O}(1)}$, for Exact k-Cycle, Longest Cycle, Longest Path, Feedback Vertex Set and Cycle Packing on unit disk graphs and unit square graphs. It is also possible to show by known NP-hardness reductions for problems on unit disk graphs [10] that an algorithm of running time $2^{o(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$ for any of our problems on unit disk graphs would imply that ETH fails. Hence our algorithms are asymptotically almost tight. Along the way we also design Turing kernels (in fact, many to one) for Exact k-Cycle, Longest Cycle, Longest Path and SI. That is, we give a polynomial time algorithm that given an instance of Exact k-Cycle or Longest Cycle or Longest Path or SI, produces polynomially many reduced instances of size polynomial in k such that the input instance is a Yes-instance if and only if one of the reduced instances is. As a byproduct of this we obtain a $2^{\mathcal{O}(k\log k)} \cdot n^{\mathcal{O}(1)}$ time algorithm for SI when G is a unit disk graph and H is an arbitrary connected graph. It is noteworthy to remark that a simple disjoint union trick implies that Exact k-Cycle, Longest Cycle, Longest Path, and SI do not admit a polynomial kernel on unit disk graphs [6]. Finally, we note that we do not use Turing kernels to design our subexponential time algorithms except for Exact k-Cycle. The subexponential time parameterized algorithm for Exact k-Cycle also uses a "double layering" of Baker's technique [3].

All our subexponential time algorithms have the following theme in common. If an input n-vertex unit disk graph G contains a clique of size $\mathsf{poly}(k)$ (such a clique can be found in polynomial time), then we have a trivial Yes-instance or No-instance, depending on the problem. Otherwise, we show that the unit disk graph G in a Yes-instance of the problem admits, sometimes after a polynomial time preprocessing, a specific type of (ω, Δ, τ)-decomposition, where the meaning of ω, Δ and τ is as follows. The vertex set of G is partitioned into cliques $C_1, \ldots, C_d$, each of size at most $\omega = k^{\mathcal{O}(1)}$. We also require that after contracting each of the cliques C_i to a single vertex, the maximum vertex degree Δ of the obtained graph $\tilde{G}$ is $\mathcal{O}(1)$, while the treewidth τ of $\tilde{G}$ is $\mathcal{O}(\sqrt{k})$. Moreover, the corresponding tree decomposition of $\tilde{G}$ can be constructed efficiently. We use the tree decomposition of $\tilde{G}$ to construct a tree decomposition of G by "uncontracting" each of the contracted cliques C_i. While the width of the obtained tree decomposition of G can be of order $\omega \cdot \tau = k^{\mathcal{O}(1)}$, we show that each of our parameterized problems can be solved in time $f(\Delta) \cdot \omega^{f(\Delta)\cdot\tau}$. Here we use dynamic programming over the constructed tree decomposition of G, however there

is a twist from the usual way of designing such algorithms. This part of the algorithm is problem-specific – in order to obtain the claimed running time, we have to establish a very specific property for each of the problems. Roughly speaking, the desired property of a problem is that it always admits an optimal solution such that for every pair of adjacent bags X, Y of the tree decomposition of G, the number of edges of this solution "crossing" a cut between X and Y is $\mathcal{O}(\sqrt{k})$. We remark that the above decomposition is *only* given in the introduction to present our ideas for all the algorithms in a unified way.

In this paper we restrict our attention to solving Exact k-Cycle and Feedback Vertex Set on unit disk graphs. We refer to the full version of the paper for all the proofs and the results.

2 Preliminaries

For a positive integer t, we use $[t]$ as a shorthand for $\{1, 2, \ldots, t\}$. Given a function $f : A \to B$ and a subset $A' \subseteq A$, let $f|_{A'}$ denote the restriction of the function f to the domain A'. For a function $f : A \to B$ and $B' \subseteq B$, $f^{-1}(B')$ denote the set $\{a \in A \ : \ f(a) \in B'\}$. For $t, t' \in \mathbb{N}$, a set $[t] \times [t']$, $i \in [t]$ and $j \in [t']$ we use $(*, j)$ and $(i, *)$ to denote the sets $\{(i', j) \ : \ i' \in [t]\}$ and $\{(i, j') \ : \ j' \in [t']\}$, respectively. For a set U, we use 2^U to denote the power set of U.

We use standard notation and terminology from the book of Diestel [16] for graph-related terms which are not explicitly defined here. Given a graph G, $V(G)$ and $E(G)$ denote its vertex-set and edge-set, respectively. When the graph G is clear from context, we denote $n = |V(G)|$ and $m = |E(G)|$. Given $U \subseteq V(G)$, we let $G[U]$ denote the subgraph of G induced by U, and we let $G \setminus U$ denote the graph $G[V(G) \setminus U]$. For an edge subset E, we use $V(E)$ to denote the set of endpoints of edges in E and $G[E]$ to denote the graph $(V(E), E)$. For $X, Y \subseteq V(G)$, we use $E(X)$ and $E(X, Y)$ to denote the edge sets $\{\{u, v\} \in E(G) \ : \ u, v \in X\}$ and $\{\{u, v\} \in E(G) \ : \ u \in X, v \in Y\}$, respectively. Moreover, we let $N(U)$ denote the open neighborhood of G. In case $U = \{v\}$, we denote $N(v) = N(U)$. Given an edge $e = \{u, v\} \in E(G)$, we use G/e to denote the graph obtained from G by contracting the edge e. In other words, G/e denotes the graph on the vertex-set $(V(G) \setminus \{u, v\}) \cup \{x_{\{u,v\}}\}$, where $x_{\{u,v\}}$ is a new vertex, and the edge-set $E(G) = E(G[V(G) \setminus \{u, v\}]) \cup \{\{x_{\{u,v\}}, w\} \mid w \in N(\{u, v\})\}$. A graph H is called a *minor* of G, if H can be obtained from G by a sequence of edge deletion, edge contraction and vertex deletion. Given $k \in \mathbb{N}$, we let K_k denote the complete graph on k vertices. For a set X, we use $K[X]$ to denote the complete graph on X. Given $a, b \in \mathbb{N}$, an $a \times b$ grid is a graph on $a \cdot b$ vertices, $v_{i,j}$ for $(i, j) \in [a] \times [b]$, such that for all $i \in [a - 1]$ and $j \in [b]$, it holds that $v_{i,j}$ and $v_{i+1,j}$ are neighbors, and for all $i \in [a]$ and $j \in [b - 1]$, it holds that $v_{i,j}$ and $v_{i,j+1}$ are neighbors. For ease of presentation, for any function $f : D \to [a] \times [b]$, $i \in [a]$ and $j \in [b]$, we use $f^{-1}(i, j)$, $f^{-1}(*, j)$, and $f^{-1}(i, *)$ to denote the sets $f^{-1}((i, j))$, $f^{-1}((*, j))$, and $f^{-1}((i, *))$, respectively.

We also need the standard notions of pathwidth, treewidth and nice tree decomposition. These definitions are given in the appendix for easy perusal (also in the appended version). However, we use slightly different but equivalent definition of path decomposition.

▶ **Definition 1.** A *path decomposition* of a graph G is a sequence $\mathcal{P} = (X_1, X_2, \ldots, X_\ell)$, where each $X_i \subseteq V(G)$ is called a *bag*, that satisfies the following conditions.

- $\bigcup_{i \in [\ell]} X_i = V(G)$.
- For every edge $\{u, v\} \in E(G)$ there exists $i \in [\ell]$ such that $\{u, v\} \subseteq X_i$.
- For every vertex $v \in V(G)$, if $v \in X_i \cap X_j$ for some $i \leq j$, then $v \in X_r$ for all $r \in \{i, \ldots, j\}$.

The *width* of $\mathcal{P}$ is $\max_{i \in [\ell]} |X_i| - 1$.

The *pathwidth* of G, $\mathsf{pw}(G)$, is the minimum width of a path decomposition of G.

▶ **Proposition 2** ([5, 7]). *Given a graph G and an integer k, in time $2^{\mathcal{O}(k)} \cdot n$, we can either decide that $\mathsf{tw}(G) > k$ or output a tree decomposition of G of width $5k$. Furthermore, given a graph G and a tree decomposition $\mathcal{T}$ of G, a nice tree decomposition $\mathcal{T}'$ of the same width as $\mathcal{T}$ can be computed in linear time.*

Given a set of geometric objects, O, we say that a graph G *represents* O if each vertex in $V(G)$ represents a distinct geometric object in O, and every geometric object in O is represented by a distinct vertex in $V(G)$. In this case, we abuse notation and write $V(G) = O$. The *intersection graph of* O is a graph G that represent O and satisfies $E(G) = \{\{u, v\} : u, v \in O, u \cap v \neq \emptyset\}$. Let $P = \{p_1 = (x_1, y_1), p_2 = (x_2, y_2), \ldots, p_n = (x_n, y_n)\}$ be a set of points in the Euclidean plane. In the *unit disk graph model*, for every $i \in [n]$, we let d_i denote the disk of radius 1 whose centre is p_i. Accordingly, we denote $D = \{d_1, d_2, \ldots, d_n\}$. Then, the *unit disk graph of* D is the intersection graph of D. Alternatively, the unit disk graph of D is the geometric graph of G such that $E(G) = \{\{p_i = (x_i, y_i), p_j = (x_j, y_j)\} \mid p_i, p_j \in D, i \neq j, \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \leq 2\}$.

3 Clique-Grid Graphs

In this section, we introduce a family of "grid-like" graphs, called clique-grid graphs, that is tailored to fit our techniques. Given a unit disk graph G, we extract the properties of G that we would like to exploit, and show that they can be captured by an appropriate clique-grid graph. Let us begin by giving the definition of a clique-grid graph. Roughly speaking, a graph G is a clique-grid graph if each of its vertices can be embedded into a single cell of a grid (where multiple vertices can be embedded into the same cell), ensuring that the subgraph induced by each cell is a clique, and that each cell can interact (via edges incident to its vertices) only with cells at "distance" at most 2. Formally,

▶ **Definition 3** (clique-grid graph). A graph G is a *clique-grid graph* if there exists a function $f : V(G) \to [t] \times [t']$, for some $t, t' \in \mathbb{N}$, such that the following conditions are satisfied.

1. For all $(i, j) \in [t] \times [t']$, it holds that $f^{-1}(i, j)$ is a clique.
2. For all $\{u, v\} \in E(G)$, it holds that if $f(u) = (i, j)$ and $f(v) = (i', j')$ then $|i - i'| \leq 2$ and $|j - j'| \leq 2$.

Such a function f is a *representation* of G.

We note that a notion similar to clique-grid graph was also used by Ito and Kadoshita [31]. For the sake of clarity, we say that a pair $(i, j) \in [t] \times [t']$ is a *cell*. The following lemma states that a unit disk graph is a clique-grid graph, and its proof is deferred to [21].

▶ **Lemma 4.** *Let D be a set of points in the Euclidean plane, and let G be the unit disk graph of D. Then, a representation f of G can be computed in polynomial time.*

Due to Lemma 4, throughout this paper, we assume that along with a input unit disk graph G, we are given a representation f of G. We conclude this section by introducing the definition of an ℓ-NCTD which is useful for doing our dynamic programming algorithms.

▶ **Definition 5.** A tree decomposition $\mathcal{T} = (T, \beta)$ of a clique-grid graph G with representation f is a *nice ℓ-clique tree decomposition*, or simply an *ℓ-NCTD*, if for the root r of T, it holds that $\beta(r) = \emptyset$, and for each node $v \in V(T)$, it holds that

- There exist at most ℓ cells, $(i_1, j_1), \ldots, (i_\ell, j_\ell)$, such that $\beta(v) = \bigcup_{t=1}^{\ell} f^{-1}(i_t, j_t)$, and
- The node v is of one of the following types.

- **Leaf**: v is a leaf in T and $\beta(v) = \emptyset$.
- **Forget**: v has exactly one child u, and there exists a cell $(i,j) \in [t] \times [t']$ such that $f^{-1}(i,j) \subseteq \beta(u)$ and $\beta(v) = \beta(u) \setminus f^{-1}(i,j)$.
- **Introduce**: v has exactly one child u, and there exists a cell $(i,j) \in [t] \times [t']$ such that $f^{-1}(i,j) \subseteq \beta(v)$ and $\beta(v) \setminus f^{-1}(i,j) = \beta(u) \setminus f^{-1}(i,j)$.
- **Join**: v has exactly two children, u and w, and $\beta(v) = \beta(u) = \beta(w)$.

A *nice ℓ-clique path decomposition*, or simply an *ℓ-NCPD*, is an ℓ-NCTD where T is a path. In this context, for convenience, we use the notation referring to a sequence presented in Section 2.

4 Turing Kernels

In this section we give an overview of a Turing kernel (actually a compression) for SI. The reason for stating our result in this way is, that this is how we use it in the next section to design a subexponential algorithm for Exact k-Cycle. We refer to the appended version for a Turing kernel for SI and Longest Cycle. More precisely, we show the following.

▶ **Theorem 6.** *Let (G, f, H, k) be an instance of* SI *on unit disk graphs, then in polynomial time, one can output a set $\mathcal{I}$ of instances of* SI *on clique-grid graphs such that (G, f, H, k) is a* Yes*-instance if and only if at least one instance in $\mathcal{I}$ is a* Yes*-instance, and for all $(\widehat{G}, \widehat{f} : V(\widehat{G}) \to [\widehat{t}] \times [\widehat{t'}], \widehat{H}, \widehat{k}) \in \mathcal{I}$, $\widehat{G}$ is either an induced subgraph of G or $K_{\widehat{k}}$, $\widehat{t}, \widehat{t'} \leq 2\widehat{k}$, $|\widehat{f}^{-1}(i,j)| < \widehat{k}$ for any $(i,j) \in [\widehat{t}] \times [\widehat{t'}]$, $\widehat{H} = H$, $\widehat{k} = k$, $|V(\widehat{G})| = \mathcal{O}(k^3)$, and $|E(\widehat{G})| = \mathcal{O}(k^4)$.*

Proof Sketch. First, suppose that there exists a cell $(i,j) \in [t] \times [t']$ such that $|f^{-1}(i,j)| \geq k$, then by Definition 3, $G[f^{-1}(i,j)]$ is a clique on at least k vertices. In particular, the pattern H is a subgraph of $G[f^{-1}(i,j)]$, and therefore it is also a subgraph of G. Thus, in this case, we conclude the proof by setting $\mathcal{I}$ to be the set that contains only one instance, $(K_k, \widehat{f} : V(K_k) \to [1] \times [1], H, k)$. From now on, suppose that for all cells $(i,j) \in [t] \times [t']$, it holds that $|f^{-1}(i,j)| < k$.

Now, our kernelization algorithm works as follows. For every $(p,q) \in [t] \times [t']$, it computes

$$G_{p,q} = G[\bigcup_{\substack{p \leq i < \min\{p+2k, t+1\} \\ q \leq j < \min\{q+2k, t'+1\}}} f^{-1}(i,j)].$$

Accordingly, it computes $f_{p,q} : V(G_{p,q}) \to [\min\{2k,t\}] \times [\min\{2k,t'\}]$ as follows. For every $v \in V(G_{p,q})$, compute $f_{p,q}(v) = (i-p+1, j-q+1)$ where $(i,j) = f(v)$. Note that for all $(i,j) \in [\min\{2k,t\}] \times [\min\{2k,t'\}]$, it holds that $f_{p,q}^{-1}(i,j) = f^{-1}(i+p-1, j+q-1)$. Thus, since f is a representation of G, it holds that $f_{p,q}$ is a representation of $G_{p,q}$. Finally, our kernelization algorithm outputs $\mathcal{I} = \{I_{p,q} = (G_{p,q}, f_{p,q}, H, k) : (p,q) \in [t] \times [t']\}$.

To conclude the proof, it remains to show that (G, f, H, k) is a Yes-instance if and only if at least one instance in $\mathcal{I}$ is a Yes-instance. Since for all $(G_{p,q}, f_{p,q}, H, k) \in \mathcal{I}$, it holds that $G_{p,q}$ is an induced subgraph of G, we have that if (G, f, H, k) is a No-instance, then every instance in $\mathcal{I}$ is No-instance as well. Next, suppose that (G, f, H, k) is a Yes-instance. Then, let H' be a subgraph of G that is isomorhpic to H. In order to complete the proof, we introduce a notion of *ℓ-stretched*. We say that H' is *ℓ-stretched* if there exist cells $(i,j), (i',j') \in [t] \times [t']$ such that the following conditions are satisfied: (i) $|i-i'| \geq 2\ell$ or $|j-j'| \geq 2\ell$ (or both); and (b) $V(H') \cap f^{-1}(i,j) \neq \emptyset$ and $V(H') \cap f^{-1}(i',j') \neq \emptyset$. One can show (see the appended version) that for any subgraph H' of G that is isomorphic to H, it holds that H' is not $2k$-stretched. Using this *claim* we can conclude. Denote $i_{\min} = \min\{i \in [t] : (\bigcup_{j \in [t']} f^{-1}(i,j)) \cap V(H') \neq \emptyset\}$,

$i_{\max} = \max\{i \in [t] : (\bigcup_{j\in[t']} f^{-1}(i,j)) \cap V(H') \neq \emptyset\}$, $j_{\min} = \min\{j \in [t'] : (\bigcup_{i\in[t]} f^{-1}(i,j)) \cap V(H') \neq \emptyset\}$ and $j_{\max} = \max\{j \in [t'] : (\bigcup_{i\in[t]} f^{-1}(i,j)) \cap V(H') \neq \emptyset\}$. From the above claim, it holds that both $i_{\max} - i_{\min} < 2k$ and $j_{\max} - j_{\min} < 2k$. Therefore, H' is a subgraph of $G_{i_{\min},j_{\min}}$, which implies that $I_{p,q}$ is a YES-instance. ◀

5 Exact k-Cycle

In this section we prove the following theorem.

▶ **Theorem 7.** EXACT k-CYCLE *on unit disk graphs can be solved in* $2^{\mathcal{O}(\sqrt{k}\log k)}n^{\mathcal{O}(1)}$ *time.*

By Theorem 6, we have seen that SI admits a polynomial sized Turing kernel. Hence to give an algorithm of running time $2^{\mathcal{O}(\sqrt{k}\log k)}|V(G)|^{\mathcal{O}(1)}$, we can restrict to instances returned by Theorem 6. More precisely, because of Theorem 6, we can assume that the input to our algorithm is $(G, f : V(G) \to [t] \times [t'], k)$ where G is a clique-grid graph with a representation f, $|f^{-1}(i,j)| < k$ for all $(i,j) \in [t] \times [t']$, and $t, t' \leq 2k$. Without loss of generality we can assume that f is a function from $V(G)$ to $[2k] \times [2k]$, because $[t] \times [t'] \subseteq [2k] \times [2k]$.

Given an instance $(G, f : V(G) \to [2k] \times [2k], k)$, the algorithm applies a method inspired by Baker's technique [3] and obtains a family, $\mathcal{F}$, of $2^{\mathcal{O}(\sqrt{k}\log k)}$ instances of EXACT k-CYCLE. The family $\mathcal{F}$ has following properties.

1. In each of these instances the input graph is an induced subgraph of G and has size $k^{\mathcal{O}(1)}$.
2. The input $(G, f : V(G) \to [2k] \times [2k], k)$ is a YES-instance if and only if there exists an instance $(H, f^* : V(H) \to [2k] \times [2k], k) \in \mathcal{F}$ which is a YES-instance.
3. More over, for any instance $(H, f^* : V(H) \to [2k] \times [2k], k) \in \mathcal{F}$, H has a nice $7\sqrt{k}$-clique path decomposition ($7\sqrt{k}$-NCPD).

We will call the family $\mathcal{F}$ satisfying the above properties as *good family*. Let $(H, f^* : V(H) \to [2k] \times [2k], k)$ be an instance of $\mathcal{F}$. Let $\mathcal{P} = (X_1, \ldots, X_q)$ be a $7\sqrt{k}$-NCPD of H. We first prove that if there is a cycle of length k in H, then there is a cycle C of length k in H such that for any two distinct cells (i,j) and (i',j') of f, the number of edges with one end point in (i,j) and other in (i',j') is at most 4. Let C be such a cycle in H. Then using the property of C we get the following important property.

> For any $i \in [q]$, the number of edges of $V(C)$ with one end point in X_i and other in $\bigcup_{i<j\leq q} X_j$ is in $\mathcal{O}(\sqrt{k})$.

The above mentioned property allows us to design a dynamic programming (DP) algorithm over $7\sqrt{k}$-NCPD, $\mathcal{P}$, for EXACT k-CYCLE in time $2^{\mathcal{O}(\sqrt{k}\log k)}$. Now we are ready to give formal details about the algorithm. As explained before, we assume that the number of vertices in the input graph is bounded by $k^{\mathcal{O}(1)}$.

▶ **Lemma 8.** *Let* $(G, f : V(G) \to [2k] \times [2k], k)$ *be an instance of* EXACT k-CYCLE*, where* G *is a clique-grid graph with representation* f*,* $|f^{-1}(i,j)| < k$ *for all* $(i,j) \in [2k] \times [2k]$ *and* $|V(G)| = k^{\mathcal{O}(1)}$*. Given* $(G, f : V(G) \to [2k] \times [2k], k)$*, there is an algorithm running in time* $2^{\mathcal{O}(\sqrt{k}\log k)}$ *that outputs a good family* $\mathcal{F}$*.*

Proof. Let C be a k length cycle in G. First we define a column of the $2k \times 2k$ grid. For any $j \in [2k]$ the set of cells $\{(i,j) \ : \ i \in [2k]\}$ is called a column. There are $2k$ columns for the $2k \times 2k$ grid. We partition $2k$ columns of the $2k \times 2k$ grid with k blocks of two consecutive columns and label them from the set of labels $[\sqrt{k}]$. That is, both columns $2i-1$ and $2i$,

where $i \in [k]$, are labelled with $i \mod \sqrt{k}$. Then by pigeon hole principle there is a label $\ell \in \{1, 2, \ldots, \sqrt{k}\}$ such that the number of vertices from $V(C)$ which are in columns labelled ℓ is at most $\sqrt{k}$. As $|V(G)| \leq k^{\mathcal{O}(1)}$, the number of vertices of G in columns labelled ℓ is at most $k^{\mathcal{O}(1)}$. We guess the vertices of $V(C)$ which are in the columns labelled ℓ. The number of potential guesses is bounded by $k^{\mathcal{O}(\sqrt{k})}$. Let Y be the set of guessed vertices of $V(C)$ which are in the columns labelled by ℓ. Notice that $|Y| \leq \sqrt{k}$. Then we delete all the vertices in columns labelled ℓ, except the vertices of Y. Let S be the set of deleted vertices. By the property 2 of clique-grid graph, $G \setminus (S \cup Y)$ is a disjoint union of clique-grid graphs each of which is represented by a function with at most $2\sqrt{k}$ columns. That is, let $G_1 = G[\bigcup_{j=1}^{2(\ell-1)} f^{-1}(*, j)]$, and $G_{i+1} = G[\bigcup_{j=i\cdot 2\ell+1}^{\min\{i\cdot 2\ell+2\sqrt{k}, 2k\}} f^{-1}(*, j)]$ for all $i \in \{1, \ldots, \lceil\sqrt{k}\rceil\}$. Notice that G_i is clique-grid graph with representation $f_i : V(G_i) \to [2k] \times [2\sqrt{k}]$ defined as, $f_i(u) = (r, j)$, when $f(u) = (r, (i-1)2\ell+j)$. By the property 2 of clique-grid graph, $G \setminus (S \cup Y) = G_1 \uplus \ldots \uplus G_{\lceil\sqrt{k}\rceil+1}$.

▶ **Claim 1.** *$G \setminus S$ has a nice $7\sqrt{k}$-clique path decomposition ($7\sqrt{k}$-NCPD).*

Proof. First, for each $i \in \{1, \ldots, \lceil\sqrt{k}\rceil + 1\}$, we will define a path decomposition of G_i (in the next paragraph) such that each bag is a union of at most $6\sqrt{k}$ many cells of f_i. As $G \setminus (S \cup Y) = G_1 \uplus \ldots \uplus G_{\lceil\sqrt{k}\rceil-1}$, and $|Y| \leq \sqrt{k}$, by adding Y to each bag of all path decompositions we can get a required nice $7\sqrt{k}$-clique path decomposition for $G \setminus S$.

Now, for each G_i, we define a path decomposition $\mathcal{P}_i = (X_{i,1}, X_{i,2}, \ldots X_{i,2k-2})$ where $X_{i,j} = f_i^{-1}(j, *) \cup f_i^{-1}(j+1, *) \cup f_i^{-1}(j+2, *)$. We claim that $\mathcal{P}_i$ is a path decomposition of G_i. Notice that $\bigcup_{j=1}^{k-1} X_{i,j} = f_i^{-1}(*, *) = V(G_i)$. By property 2 of clique-grid graph, we have that for each edge $\{u, v\} \in E(G)$, there exists $j \in [2k-2]$ such that $\{u, v\} \in X_{i,j}$. For each $u \in V(G)$, u is contained in at most three bags and these bags are consecutive in the sequence $(X_{i,1}, X_{i,2}, \ldots X_{i,2k-2})$. Hence $\mathcal{P}_i$ is a path decomposition of G_i. Since $X_{i,j} = f_i^{-1}(j, *) \cup f_i^{-1}(j+1, *) \cup f_i^{-1}(j+2, *)$, number of columns in f_i is at most $2\sqrt{k}$ and each cell of f_i is a cell of f, each $X_{i,j}$ is a union of $6\sqrt{k}$ many cells of f. Since $G \setminus (S \cup Y) = G_1 \uplus \ldots \uplus G_{\lceil\sqrt{k}\rceil+1}$, the sequence $\mathcal{P}' = (X_{1,1}, \ldots X_{1,2k-2}, X_{2,1}, \ldots X_{2,2k-2}, \ldots, X_{\lceil\sqrt{k}\rceil-1,1}, \ldots X_{\lceil\sqrt{k}\rceil-1,2k-2})$ is a path decomposition of $G \setminus (S \cup Y)$. More over, the vertices of each bag is a union of vertices from at most $6\sqrt{k}$ cells of f. Also, since $|Y| \leq \sqrt{k}$, the sequence $\mathcal{P} = (X_{1,1} \cup Y, \ldots X_{1,k-2} \cup Y, \ldots X_{\lceil\sqrt{k}\rceil-1,k-2} \cup Y)$ obtained by adding Y to each bag of $\mathcal{P}'$ we get a path decomposition of $G \setminus S$. More over, the vertices of each bag in $\mathcal{P}$ is a union of vertices from at most $7\sqrt{k}$ cells of f. We can turn the path decomposition $\mathcal{P}$ to a $7\sqrt{k}$-NCPD by an algorithm similar to the one mentioned in Proposition 2. ◀

The algorithm constructs a family $\mathcal{F}$ as follows. For each $\ell \in \{1, \ldots, \lceil\sqrt{k}\rceil\}$ and for two subsets of vertices S and Y such that $S \cup Y$ is a set of vertices in the columns labelled ℓ and $|Y| \leq \sqrt{k}$, our algorithm includes an instance $(G \setminus S, f|_{V(G)\setminus S}, k)$ in $\mathcal{F}$. The number of choices of S and Y is bounded by $2^{\mathcal{O}(\sqrt{k}\log k)}$ and thus the size of $\mathcal{F}$ is bounded by $2^{\mathcal{O}(\sqrt{k}\log k)}$. We claim that $\mathcal{F}$ is indeed a good family. Let C be a cycle of length k in G. Then, there is an $\ell \in \{1, \ldots, \lceil\sqrt{k}\rceil\}$ such that at most $\sqrt{k}$ vertices from $V(C)$ are in the columns labelled by ℓ. Let S' be the set of vertices in the columns labelled by ℓ. Let $Y = S' \cap V(C)$ and $S = S' \setminus Y$. The instance $(G \setminus S, f|_{V(G)\setminus S}, k)$ in $\mathcal{F}$ is a YES instance. This concludes the proof. ◀

Now we can assume that we are solving EXACT k-CYCLE on (H, f, k), where $(H, f, k) \in \mathcal{F}$ (here we rename the function $f|_{V(H)}$ with f for ease of presentation). Now we prove that if there is a cycle of length k in H, then there is a cycle C of length k in H such that for any two cells (i, j) and (i', j') of f, the number of edges of $E(C)$ with one end point in (i, j) and other (i', j') is at most 5.

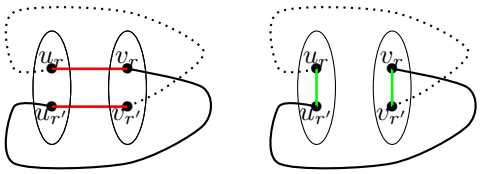

Figure 1 Illustration of Lemma 9. Figure on the left is the cycle $C = [u_r v_r]Q_1[u_{r'} v_{r'}]Q_2$ and the one on the right is the cycle $C' = [u_r u_{r'}]\overleftarrow{Q}_1[v_r v_{r'}]Q_2$.

▶ **Lemma 9.** *Let $(H, f : V(H) \to [2k] \times [2k], k)$ be a* Yes *instance of* Exact k-Cycle. *Then there is a cycle C of length k in H such that for any two distinct cells (i, j) and (i', j') of f, the number of edges of $E(C)$ with one end point in (i, j) and other (i', j') is at most* 5.

Proof. Let C be a k length cycle such that the number edges of $E(C)$ whose end points are in different cells is minimized. We claim that for any two disjoint cells (i, j) and (i', j'), the number of edges of $E(C)$ with one end point in (i, j) and other (i', j') is at most 5. Suppose not. Then there exist (i, j) and (i', j') such that the number of edges of $E(C)$ with one end point in (i, j) and other in (i', j') is at least 6. Let $C = P_1[u_1 v_1]P_2[u_2 v_2]P_3[u_3 v_3]P_4[u_4 v_4]P_5[u_5 v_5]P_6[u_6 v_6]$ where for each $\{u_r, v_r\}, r \in [6]$, one end point is in the cell (i, j) and other in the cell $(i'j')$, and each subpath $P_\ell, \ell \in [6]$, can be empty too. Since C is a cycle, at least 3 edges from $\{\{u_r, v_r\} : i \in [6]\}$ form a matching. Let $\{u_{r_1}, v_{r_1}\}, \{u_{r_2}, v_{r_2}\}$ and $\{u_{r_3}, v_{r_3}\}$ be a matching of size 3, where $\{r_1, r_2, r_3\} \subseteq [6]$. Then, by pigeon hole principle there exist $r, r' \in \{r_1, r_2, r_3\}$ such that either $u_r, u_{r'} \in f^{-1}(i, j)$ or $u_r, u_{r'} \in f^{-1}(i', j')$. Without loss of generality assume that $u_r, u_{r'} \in f^{-1}(i, j)$ (otherwise we rename cell (i, j) with (i', j') and vice versa). That is, $C = [u_r v_r]Q_1[u_{r'} v_{r'}]Q_2$ such that $u_r, u_{r'} \in f^{-1}(i, j)$ and $v_r, v_{r'} \in f^{-1}(i', j')$. Then, since $f^{-1}(i, j)$ and $f^{-1}(i', j')$ are cliques, $C' = [u_r u_{r'}]\overleftarrow{Q}_1[v_r v_{r'}]Q_2$ is a k length cycle in G, such that the number edges of $E(C')$ whose end points are in different cells is less than that of $E(C)$, which is contradiction to our assumption. See Fig. 1 for an illustration of C and C'. This completes the proof. ◀

Next we design a DP algorithm that finds a cycle of length k, if it exists, satisfying properties of Lemma 9.

▶ **Lemma 10.** *Let $(H, f : V(H) \to [2k] \times [2k], k) \in \mathcal{F}$ be an instance of* Exact k-Cycle. *and $\mathcal{P}$ be a $7\sqrt{k}$-NCPD of H. Then, given $(H, f : V(H) \to [2k] \times [2k], k)$ and $\mathcal{P}$, there is an algorithm $\mathcal{A}$ which runs in time $2^{\mathcal{O}(\sqrt{k} \log k)}$, and outputs* Yes, *if there is a cycle C in H such that for any two distinct cells (i, j) and (i', j') of f, the number of edges with one end point in (i, j) and other (i', j') is at most* 5. *Otherwise algorithm $\mathcal{A}$ will output* No.

Proof Sketch. Algorithm $\mathcal{A}$ is a DP algorithm over the $7\sqrt{k}$-NCPD $\mathcal{P} = (X_1, \ldots X_q)$ of H. For any $\ell \in [q]$, we define H_ℓ be the induced subgraph $H[\bigcup_{i \leq \ell} X_i]$ of H. Define $\mathcal{C}$ to be the set of k-length cycles in H such that for any $C \in \mathcal{C}$ and two disjoint cells (i, j) and (i', j') of f, the number of edges of $E(C)$ with one end point in (i, j) and other (i', j') is at most 5. Let $C \in \mathcal{C}$. Since $\mathcal{P}$ is a $7\sqrt{k}$-NCPD and the fact that for any two distinct cells (i, j) and (i', j') of f, the number of edges of C with one end point in (i, j) and other (i', j') is at most 5, we have that for any bag X_ℓ of $\mathcal{P}$, the number of vertices of $V(C) \cap X_\ell$ which has a neighbour in $V(H) \setminus X_\ell$ is bounded by $\mathcal{O}(\sqrt{k})$. This allows us to keep only $2^{\mathcal{O}(\sqrt{k} \log k)}$ states in the DP algorithm. Fix any $\ell \in [q]$ and define C_L the set of paths of C (or the cycle C itself) when we restrict C to H_ℓ. That is $C_L = H_\ell[E(C)]$. Let $\widehat{C}_L = \{\{u, v\} \mid \text{there is a } u\text{-}v \text{ path in } C_L\}$. Notice that $\bigcup_{P \in \widehat{C}_L} P$ is the set of vertices of degree 0 or 1 in C_L and $\bigcup_{P \in \widehat{C}_L} P \subseteq X_\ell$. Since

X_ℓ is a union of vertices from at most $7\sqrt{k}$ many cells of f and for any two distinct cells (i,j) and (i',j') of f, the number of edges of $E(C)$ with one end point in (i,j) and other (i',j') is at most 5, and by property 2 of the clique-grid graph, we have that the cardinality of $\bigcup_{P \in \widehat{C}_L} P$ is at most $5 \cdot 24 \cdot 7\sqrt{k} = 840\sqrt{k}$. In our DP algorithm we will have state indexed by $(\ell, \widehat{C}_L, |E(C_L)|)$ which will be set to 1. Formally, for any $\ell \in [q]$, $k' \in [k]$ and a family $\mathcal{Z}$ of vertex disjoint sets of size at most 2 of X_ℓ with the property that the cardinality of $\bigcup_{Z \in \mathcal{Z}} Z$ is at most $840\sqrt{k}$, we will have a DP table entry $\mathcal{A}[\ell, \mathcal{Z}, k']$. For each $\ell \in [q]$, we maintain the following correctness invariant.

Correctness Invariants: (i) For every $C \in \mathcal{C}$, let $C_L = H_\ell[E(C)]$. For every $C \in \mathcal{C}$, let $\widehat{C}_L = \{\{u,v\} \mid$ there is a connected component P in C_L and P is a u-v path as well$\}$. Then $\mathcal{A}[\ell, \widehat{C}_L, |E(C_L)|] = 1$, (ii) for any family $\mathcal{Z}$ of vertex disjoint sets of size at most 2 of X_ℓ with $0 < |\bigcup_{Z \in \mathcal{Z}} Z| \leq 840\sqrt{k}$, $k' \in [k]$, and $\mathcal{A}[\ell, \mathcal{Z}, k'] = 1$, there is a set $\mathcal{Q}$ of $|\mathcal{Z}|$ vertex disjoint paths in H_ℓ where the end points of each path are specified by a set in $\mathcal{Z}$ and $|E(\mathcal{Q})| = k'$, and (iii) if $\mathcal{A}[\ell, \emptyset, k] = 1$, then there is a cycle of length k in H_ℓ.

The correctness of the our algorithm will follow from the correctness invariant. We fill the DP table entries similar to the way it is done for dynamic programming algorithms on graphs of bounded treewidth. That is, we fill the table entries by considering various cases for bags (introduce and forget) and using the previously computed DP table entries. A complete detailed proof is provided in the appended version. ◀

Theorem 7 follows from Theorem 6 and Lemmata 8, 9, and 10.

6 Feedback Vertex Set

In this section, we show that Feedback Vertex Set (FVS) admits a subexponential-time parameterized algorithm. More precisely, we prove the following.

▶ **Theorem 11.** FVS *on unit disk graphs can be solved in time* $2^{\mathcal{O}(\sqrt{k}\log k)}n^{\mathcal{O}(1)}$.

For an algorithm for FVS and other problems such as Cycle Packing, we need more compact representations of clique-grid graphs. In the next section we introduce these notions.

6.1 The Cell Graph of a Clique-Grid Graph

We introduce two compact representations of clique-grid graphs. By examining these representations, we are able to infer information on the structure of clique-grid graphs that are also unit disk graphs.

▶ **Definition 12** (backbone)**.** Given a clique-grid graph G with representation $f : V(G) \to [t] \times [t']$, an induced subgraph H of G is a *backbone* for (G, f) if for any two distinct cells $(i,j), (i',j') \in [t] \times [t']$ for which there exist $u \in f^{-1}(i,j)$ and $v \in f^{-1}(i',j')$ such that $\{u,v\} \in E(G)$, there also exist $u' \in f^{-1}(i,j)$ and $v' \in f^{-1}(i',j')$ such that $\{u',v'\} \in E(H)$. If no induced subgraph of H is a backbone for (G,f), then H is a *minimal backbone* for (G,f).

First, we bound the maximum degree of a minimal backbone.

▶ **Lemma 13.** *Let G be a clique-grid graph with representation f, and let H be a minimal backbone for (G,f). Then, for all $(i,j) \in [t] \times [t']$, it holds that $|f^{-1}(i,j) \cap V(H)| \leq 24$. Furthermore, the maximum degree of H is at most* 599.

Proof. By Condition 2 in Definition 3, we have that for all cells $(i,j) \in [t] \times [t']$, it holds that $f^{-1}(i,j) \cap V(H) \leq |\{(i',j') \in [t] \times [t'] \setminus \{(i,j)\} \mid |i-i'| \leq 2, |j-j'| \leq 2\}| \leq 24$. Thus, for all $(i,j) \in [t] \times [t']$, the degree in H of a vertex in $f^{-1}(i,j) \cap V(H)$ is bounded by $|(\bigcup_{\{(i',j') \in [t] \times [t'] \mid |i-i'| \leq 2, |j-j'| \leq 2\}} f^{-1}(i,j) \cap V(H)) \setminus \{v\}| \leq |\{(i',j') \in [t] \times [t'] \mid |i-i'| \leq 2, |j-j'| \leq 2\}| \cdot 24 - 1 = 25 \cdot 24 - 1 = 599$. ◀

We compute a minimal backbone as follows. Initializes $H = G$; then, for every vertex $v \in V(G)$, it checks if the graph $H \setminus \{v\}$ has the same backbone as H, in which case it updates H to $H \setminus \{v\}$. Thus, a minimal backbone H for (G,f) can be computed in polynomial time. To analyze the treewidth of a backbone, we need the following.

▶ **Proposition 14** ([23]). *Let G be a unit disk graph with maximum degree Δ. Then G contains a $\frac{\mathsf{tw}}{100\Delta^3} \times \frac{\mathsf{tw}}{100\Delta^3}$ grid as a minor.*

▶ **Lemma 15.** *Given a clique-grid graph G that is a unit disk graph, a representation f of G and an integer $\ell \in \mathbb{N}$, in time $2^{\mathcal{O}(\ell)} \cdot n^{\mathcal{O}(1)}$, one can either correctly conclude that G contains a $\frac{\ell}{100 \cdot 599^3} \times \frac{\ell}{100 \cdot 599^3}$ grid as a minor, or obtain a minimal backbone H for (G,f) with a nice tree decomposition $\mathcal{T}$ of width at most 5ℓ.*

We will use Lemma 15 with $\ell = \mathcal{O}(\sqrt{k})$. Next, we define a more compact representation of a clique-grid graph.

▶ **Definition 16** (cell graph). Given a clique-grid graph G with representation $f : V(G) \to [t] \times [t']$, the *cell graph* of G, denoted by $\mathsf{cell}(G)$, is the graph on the vertex-set $\{v_{i,j} : i \in [t], j \in [t'], f^{-1}(i,j) \neq \emptyset\}$ and edge-set $\{\{v_{i,j}, v_{i',j'}\} : (i,j) \neq (i',j'), \exists u \in f^{-1}(i,j) \exists v \in f^{-1}(i',j')$ such that $\{u,v\} \in E(G)\}$.

By Definitions 12 and 16, and the that for any graph G and a minor H of G, it holds that $\mathsf{tw}(H) \leq \mathsf{tw}(G)$, we conclude the following.

▶ **Observation 17.** *For a clique-grid graph G, a representation f of G and a backbone H for (G,f), it holds that $\mathsf{cell}(G)$ is a minor of H and $\mathsf{tw}(\mathsf{cell}(G)) \leq \mathsf{tw}(H)$.*

Note that a nice tree decomposition of $\mathsf{cell}(G)$ of width 5ℓ corresponds to a 5ℓ-NCTD of G. In other words, from Lemma 15 and Observation 17, we directly have the following..

▶ **Corollary 18.** *Given a clique-grid graph G that is a unit disk graph, a representation f of G and an integer $\ell \in \mathbb{N}$, in time $2^{\mathcal{O}(\ell)} \cdot n^{\mathcal{O}(1)}$, one can either correctly conclude that G contains a $\frac{\ell}{100 \cdot 599^3} \times \frac{\ell}{100 \cdot 599^3}$ grid as a minor, or compute a 5ℓ-NCTD of G.*

6.2 Outline of an Algorithm for FVS

First, we observe that if we find a large grid, we can answer No (see also [15, 12]).

▶ **Observation 19.** *Let (G,k) be an instance of* FVS. *If G contains a $2\sqrt{k} \times 2\sqrt{k}$ grid as a minor, then (G,k) is a* No*-instance.*

This observation leads us to the following.

▶ **Lemma 20.** *Let (G,O,k) be an instance of* FVS *on unit disk graphs. Then, in time $2^{\mathcal{O}(\sqrt{k}\log k)} \cdot |V(G)|^{\mathcal{O}(1)}$, one can either solve (G,O,k) or obtain an equivalent instance (G,f,k) of* FVS *on clique-grid graphs together with an $\mathcal{O}(\sqrt{k})$-NCTD of G.*

Proof. First, by using Lemma 4, we obtain a representation f of G. Then, by using Corollary 18 with $\ell = 200 \cdot 599^3 \cdot \sqrt{k} = \mathcal{O}(\sqrt{k})$, we either correctly conclude that G contains a $2\sqrt{k} \times 2\sqrt{k}$ grid as a minor, or compute an $\mathcal{O}(\sqrt{k})$-NCTD of G. In both cases, by Observation 19, we are done. ◀

Because of Lemma 20, to prove Theorem 11, we can focus on FVS on clique-grid graphs, where the input also contains a $\mathcal{O}(\sqrt{k})$-NCTD. That is, the input of FVS on clique-grid graphs is a tuple $(G, f, k, \mathcal{T})$ where G is a clique-grid graph with representation f and $\mathcal{T} = (T, \beta)$ is a $\mathcal{O}(\sqrt{k})$-NCTD of G. Notice that if there is a cell (i, j) of f, such that $|f^{-1}(i, j)| \geq k + 3$, then there is no feedback vertex set of size at most k in G, because $f^{-1}(i, j)$ is a clique of size at least $k + 3$ in G.

▶ **Observation 21.** *Let $(G, f, k, \mathcal{T})$ be an instance of* FVS*, where G is a clique-grid graph with representation f. If there is a cell (i, j) in f such that $|f^{-1}(i, j)| \geq k+3$, then $(G, f, k, \mathcal{T})$ is a* No*-instance.*

The following observation follows from the fact that $\mathcal{T} = (T, \beta)$ is a $\mathcal{O}(\sqrt{k})$-NCTD and $|f^{-1}(i, j)| \leq k + 2$ for any cell (i, j) of f.

▶ **Observation 22.** *For any $v \in V(T)$, $|\beta(v)| = \mathcal{O}(k^{1.5})$.*

Notice that G has a feedback vertex set of size at most k if and only if there is a vertex subset $F \subseteq V(G)$ of cardinality at least $|V(G)| - k$ such that $G[F]$ is a forest. Hence, instead of stating the problem as finding a k sized feedback vertex set in G, we can state it as finding an induced subgraph H of G with maximum number of vertices such that H is a forest.

Max Induced Forest (MIF) **Parameter:** k
Input: A clique-grid graph G with representation f and an integer k such that $\mathcal{T}$ is a $c\sqrt{k}$-NCTD of G and for any cell (i, j) in f, $\lvert f^{-1}(i, j)\rvert \leq k + 2$, where c is a constant
Question: Is there subset $W \subseteq V(G)$ such that $G[W]$ is a forest and $\lvert W\rvert \geq \lvert V(G)\rvert - k$

▶ **Observation 23.** *Let $(G, f, k, \mathcal{T})$ be an instance of* MIF*. Then $(G, f, k, \mathcal{T})$ is a* Yes*-instance of* MIF *if and only if $(G, f, k, \mathcal{T})$ is a* Yes*-instance of* FVS*.*

By Lemma 20 and Observations 21 and 23, to prove Theorem 11, it is sufficient that we prove the following result (which is the focus of the rest of this section).

▶ **Lemma 24.** MIF *on clique-grid graphs can be solved in time $2^{\mathcal{O}(\sqrt{k} \log k)} \cdot n^{\mathcal{O}(1)}$.*

Proof sketch. We explain a DP algorithm which given as input $(G, f, k, \mathcal{T})$ where G is a clique-grid graph with representation f, $\mathcal{T} = (T, \beta)$ is a $c\sqrt{k}$-NCTD, c is a constant and $|f^{-1}(i, j)| \leq k + 2$ for any cell (i, j) of f and outputs Yes if there is an induced forest with at least $|V(G)| - k$ vertices and outputs No otherwise. Here we use the term solution for a vertex subset $S \subseteq V(G)$ with the property that $G[S]$ is a forest. First notice that any solution S contains at most 2 vertices from $f^{-1}(i, j)$ for any cell (i, j). Now, the following claim follows from the fact that $\mathcal{T}$ is a $c\sqrt{k}$-NCTD and any solution contain at most 2 vertices from $f^{-1}(i, j)$ for any cell (i, j).

▶ **Claim 2.** *For any $v \in V(T)$ and any solution S, $|S \cap \beta(v)| \leq 2c\sqrt{k}$.*

We first briefly explain what is the table entries in a standard DP algorithm for our problem on graphs of bounded treewidth [12]. Then we explain that in fact many of the entries we compute in the standard DP table is redundant in our case, because of Observation 22 and Claim 2. That is, Observation 22 and Claim 2, shows that only $2^{\mathcal{O}(\sqrt{k} \log k)} |V(G)|^{\mathcal{O}(1)}$ many

states in the DP table are relevant in our case. Recall that for any $v \in V(T)$, $\gamma(v)$ denote the union of the bags of v and its descendants. The standard DP table for our problem will have an entry indexed by $(v, U, U_1 \uplus U_2 \ldots U_\ell = U)$ where $v \in V(T)$, $U \subseteq \beta(v)$. The table entry $\mathcal{A}[v, U, U_1 \uplus U_2 \ldots U_\ell]$ stores the following information: the maximum cardinality of a vertex subset $W \subseteq G[\gamma(v)]$ such that $W \cap \beta(v) = U$, $G[W]$ is a forest with a set of connected components $\mathcal{C}$ and for any $C \in \mathcal{C}$, either $V(C) \cap \beta(v) = \emptyset$ or $V(C) \cap \beta(v) = U_i$ for some $i \in [\ell]$. Notice that the total number of DP table entries is bounded by $\mathsf{tw}^{\mathcal{O}(\mathsf{tw})}|V(G)|^{\mathcal{O}(1)}$ where tw is the width of the tree decomposition $\mathcal{T}$. One can easily show that the computation of the DP table at a node can be done in time polynomial in the size of the tables of its children.

By Observation 22 and Claim 2, we know that for any bag $\beta(v)$ in $\mathcal{T}$, the potential number of subsets of $\beta(v)$ which can be part of any solution is at most $2^{\mathcal{O}(\sqrt{k}\log k)}$. This implies that we only need to compute the DP table entries for indices $(v, U, U_1 \uplus U_2 \ldots U_\ell = U)$ where $v \in V(T)$, $U \subseteq \beta(v)$ and $|U| \leq 2c\sqrt{k}$. Thus, the size of DP table, and hence the time to compute it takes $2^{\mathcal{O}(\sqrt{k}\log k)}n^{\mathcal{O}(1)}$ time. This concludes the description. ◀

References

1. Jochen Alber and Jiří Fiala. Geometric separation and exact solutions for the parameterized independent set problem on disk graphs. In *Foundations of Information Technology in the Era of Network and Mobile Computing*, pages 26–37. Springer, 2002.
2. Noga Alon, Raphael Yuster, and Uri Zwick. Color-coding. *J. Assoc. Comput. Mach.*, 42(4):844–856, 1995.
3. Brenda S. Baker. Approximation algorithms for NP-complete problems on planar graphs. *J. Assoc. Comput. Mach.*, 41(1):153–180, 1994.
4. Andreas Björklund, Thore Husfeldt, Petteri Kaski, and Mikko Koivisto. Narrow sieves for parameterized paths and packings. *CoRR*, abs/1007.1161, 2010.
5. Hans L. Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM J. Comput.*, 25(6):1305–1317, 1996.
6. Hans L. Bodlaender, Rodney G. Downey, Michael R. Fellows, and Danny Hermelin. On problems without polynomial kernels. *J. Comput. Syst. Sci.*, 75(8):423–434, 2009. doi:10.1016/j.jcss.2009.04.001.
7. Hans L. Bodlaender, Pål Grønås Drange, Markus S. Dregi, Fedor V. Fomin, Daniel Lokshtanov, and Michal Pilipczuk. A $c^k n$ 5-approximation algorithm for treewidth. *SIAM J. Comput.*, 45(2):317–378, 2016. doi:10.1137/130947374.
8. Timothy M. Chan. Polynomial-time approximation schemes for packing and piercing fat objects. *J. Algorithms*, 46(2):178–189, 2003. doi:10.1016/S0196-6774(02)00294-8.
9. Jianer Chen, Iyad A. Kanj, and Weijia Jia. Vertex cover: further observations and further improvements. *Journal of Algorithms*, 41(2):280–301, 2001.
10. Brent N. Clark, Charles J. Colbourn, and David S. Johnson. Unit disk graphs. *Discrete Mathematics*, 86(1-3):165–177, 1990.
11. Kenneth L. Clarkson and Kasturi R. Varadarajan. Improved approximation algorithms for geometric set cover. *Discrete & Computational Geometry*, 37(1):43–58, 2007. doi:10.1007/s00454-006-1273-8.
12. Marek Cygan, Fedor V. Fomin, Łukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015.
13. Erik D. Demaine, Fedor V. Fomin, Mohammadtaghi Hajiaghayi, and Dimitrios M. Thilikos. Subexponential parameterized algorithms on graphs of bounded genus and H-minor-free graphs. *Journal of the ACM*, 52(6):866–893, 2005.
14. Erik D. Demaine and MohammadTaghi Hajiaghayi. Bidimensionality: new connections between FPT algorithms and PTASs. In *Proceedings of the 16th Annual ACM-SIAM*

Symposium on Discrete Algorithms (SODA 2005), pages 590–601, New York, 2005. ACM-SIAM.

15 Erik D. Demaine and MohammadTaghi Hajiaghayi. The bidimensionality theory and its algorithmic applications. *Comput. J.*, 51(3):292–302, 2008. doi:10.1093/comjnl/bxm033.

16 Reinhard Diestel. *Graph Theory, 4th Edition*, volume 173 of *Graduate texts in mathematics*. Springer, 2012.

17 Frederic Dorn, Eelko Penninkx, Hans L. Bodlaender, and Fedor V. Fomin. Efficient exact algorithms on planar graphs: Exploiting sphere cut decompositions. *Algorithmica*, 58(3):790–810, 2010. doi:10.1007/s00453-009-9296-1.

18 Adrian Dumitrescu and János Pach. Minimum clique partition in unit disk graphs. *Graphs and Combinatorics*, 27(3):399–411, 2011.

19 Fedor V. Fomin, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. Subexponential parameterized algorithms for planar and apex-minor-free graphs via low treewidth pattern covering. In *Proceedings of the 57th Annual Symposium on Foundations of Computer Science (FOCS), to appear*, 2016.

20 Fedor V. Fomin, Daniel Lokshtanov, Fahad Panolan, and Saket Saurabh. Efficient computation of representative families with applications in parameterized and exact algorithms. *Journal of the ACM*, 63(4):29, 2016. doi:10.1145/2886094.

21 Fedor V. Fomin, Daniel Lokshtanov, Fahad Panolan, Saket Saurabh, and Meirav Zehavi. Finding, Hitting and Packing Cycles in Subexponential Time on Unit Disk Graphs. *ArXiv e-prints*, April 2017. URL: https://arxiv.org/abs/1704.07279.

22 Fedor V. Fomin, Daniel Lokshtanov, Venkatesh Raman, and Saket Saurabh. Bidimensionality and EPTAS. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011, San Francisco, California, USA, January 23-25, 2011*, pages 748–759, 2011.

23 Fedor V. Fomin, Daniel Lokshtanov, and Saket Saurabh. Bidimensionality and geometric graphs. In *SODA'12 Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete algorithms*, pages 1563–1575. SIAM, 2012.

24 F. V. Fomin, D. Lokshtanov, S. Saurabh, and D. M. Thilikos. Bidimensionality and kernels. In *Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2010)*, pages 503–510. SIAM, 2010.

25 William K. Hale. Frequency assignment: Theory and applications. *Proceedings of the IEEE*, 68(12):1497–1514, 1980.

26 Sariel Har-Peled and Mira Lee. Weighted geometric set cover problems revisited. *JoCG*, 3(1):65–85, 2012.

27 Sariel Har-Peled and Kent Quanrud. Approximation algorithms for polynomial-expansion and low-density graphs. In *Algorithms – ESA 2015 – 23rd Annual European Symposium, Patras, Greece, September 14-16, 2015, Proceedings*, volume 9294, pages 717–728. Springer, 2015.

28 Dorit S. Hochbaum and Wolfgang Maass. Approximation schemes for covering and packing problems in image processing and VLSI. *Journal of the ACM*, 32(1):130–136, 1985.

29 Harry B. Hunt III, Madhav V. Marathe, Venkatesh Radhakrishnan, S. S. Ravi, Daniel J. Rosenkrantz, and Richard Edwin Stearns. NC-approximation schemes for NP- and PSPACE-hard problems for geometric graphs. *J. Algorithms*, 26(2):238–274, 1998.

30 Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity. *Journal of Computer and System Sciences*, 63(4):512–530, 2001.

31 Hiro Ito and Masakazu Kadoshita. Tractability and intractability of problems on unit disk graphs parameterized by domain area. In *Proceedings of the 9th International Symposium on Operations Research and Its Applications (ISORA10)*, pages 120–127, 2010.

32 Bart M. P. Jansen. Polynomial kernels for hard problems on disk graphs. In *Proceedings of the 12th Scandinavian Symposium and Workshops on Algorithm Theory (SWAT)*, volume 6139, pages 310–321. Springer, 2010.

33 Karl Kammerlander. C 900 – An advanced mobile radio telephone system with optimum frequency utilization. *IEEE journal on selected areas in communications*, 2(4):589–597, 1984.

34 Ioannis Koutis. Faster algebraic algorithms for path and packing problems. In *Proceedings of the 35th International Colloquium on Automata, Languages and Programming (ICALP 2008)*, volume 5125 of *Lecture Notes in Computer Science*, pages 575–586, 2008.

35 Ioannis Koutis and Ryan Williams. Algebraic fingerprints for faster algorithms. *Commun. ACM*, 59(1):98–105, 2016. doi:10.1145/2742544.

36 Dániel Marx. Efficient approximation schemes for geometric problems? In *Proceedings of the 13th Annual European Symposium on Algorithms (ESA)*, volume 3669, pages 448–459. Springer, 2005.

37 Nabil H. Mustafa, Rajiv Raman, and Saurabh Ray. Settling the apx-hardness status for geometric set cover. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages 541–550. IEEE Computer Society, 2014. doi:10.1109/FOCS.2014.64.

38 Warren D. Smith and Nicholas C. Wormald. Geometric separator theorems & applications. In *Proceedings of the 39th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 232–243. IEEE Computer Society, 1998.

39 Stéphan Thomassé. A quadratic kernel for feedback vertex set. *ACM Transactions on Algorithms*, 6(2), 2010.

40 DW Wang and Yue-Sun Kuo. A study on two geometric location problems. *Information processing letters*, 28(6):281–286, 1988.

41 Ryan Williams. Finding paths of length k in $O^*(2^k)$ time. *Inf. Process. Lett.*, 109(6):315–318, 2009.

42 Yu-Shuan Yeh, J. Wilson, and S. Schwartz. Outage probability in mobile telephony with directive antennas and macrodiversity. *IEEE journal on selected areas in communications*, 2(4):507–511, 1984.

43 Meirav Zehavi. Mixing color coding-related techniques. In *Algorithms – ESA 2015 – 23rd Annual European Symposium, Patras, Greece, September 14-16, 2015, Proceedings*, pages 1037–1049, 2015. doi:10.1007/978-3-662-48350-3_86.